Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: **Informatyka Techniczna (ITE)**
Specjalność: **Inżynieria Systemów Informatycznych (INS)**

PRACA DYPLOMOWA MAGISTERSKA

Wykorzystanie algorytmów genetycznych w systemach wykrywania intruzów w sieciach komputerowych

inż. Bartosz Błyszcz

Opiekun pracy
dr inż. Tomasz Babczyński

Słowa kluczowe: 3-6 słów

WROCŁAW, 2024

Streszczenie

Wykaz skrótów

Tabela 1. Tabela skrótów
Źródło: opracowanie własne

GA	*Genetic Algorithm*	Algorytm Genetyczny
GP	*Genetic Programming*	Programowanie Genetyczne
GNB	*Gaussian Naive Bayes*	Naiwny Klasyfikator Bayesa wykorzystujący rozkład Gaussa
ANN	*Artificial Neural Network*	Sztuczna sieć neuronowa
CNN	*Convolutional Neural Network*	Konwolucyjna sieć neuronowa
ML	*Machine Learning*	Uczenie maszynowe
AI	*Artificial Intelligence*	Sztuczna Inteligencja
IDS	*Intrusion Detection System*	System Wykrywania Intruzów
SVM	*Support Vector Machine*	Maszyna Wektorów Nośnych
AUC	*Area Under Roc Curve*	Przestrzeń pod krzywą ROC
LCDPs	*Low-code Development Platforms*	Platforma Low-code
BI	*Business Intelligence*	Narzędzia biznesowe do przekształcania danych

Spis treści

1. Wstęp

1.1. Wprowadzenie i uzasadnienie tematu pracy

Klasyfikacja danych tabelarycznych jest zagadnieniem, które na codzień dostarcza wyzwań jej twórcom z powodu mnogości danych, a także mnogości cech, a także z nierzadko małą ilością próbek. Jednym z problemów jest między innymi dobór odpowiedniego algorytmu do problemu. Dane tebalryczne występują w każdej dziedzinie, przez co raz na jakiś czas proponowane są nowe rozwiązania i algorytmy mające rozwiązać problem klasyfikacji w sposób lepszy i wydajny. Częśc twórców próbuje podchodzić do tego w sposób innowacyjny, lecz nie zawsze to wychodzi z powodu chociażby dosotoswania algorytmu pod konkretną strukturę danych, co powoduje problemy z wykorzystaniem rozwiązania dla innych danych.

Obecnie jednymi z najpopularniejszych algorytmów do klasyfikacji danych są logiczna regresja(*ang. logistic regression*), drzewo decyzyjne(*ang. decision tree*), losowy las(*ang. random forest*), maszyna wektorów nośnych(*ang. support vector machine*), naiwny bayes(*ang. Naive Bayes*). Dlatego też bardzo ważne jest porównanie wytworzonego wcześniej rozwiązania z grupą innych algorymów, które próbują przetworzyć ten sam zestaw danych.

W dzisiejszych czasach próba taka jest bardzo uproszczona chociażby przez takie platformy jak *Machine Learning Studio*, które pozwalają na wykorzystanie mocy obliczeniowej sklasteryzowanych jednostek wirtualnych do wykonywania obliczeń na odpowiednich maszynach wirtualnych, a także do budowania skomplikowanych zautomatyzowanych procesów złożonych z wielu zadań(*ang. pipeline*). W związku z czym możliwośc wykorzystania platformy chmurowej pozwoli na zautomatyzowanie procesu porównawczego oraz oddelegowanie zadań od chmury obliczeniowej co pozwoli na uniezależnienie powodzenia doświadczenia od mocy obliczeniowej komputera lokalnego, a także na ukazanie całościowo procesu porównania algorytmów klasyfikacyjnych.

1.2. Cel pracy dyplomowej

Celem niniejszej pracy dyplomowej jest porównanie algorytmu klasyfikacji danych tabelarycznych wypracowanego w trakcie pisania pracy inżynierskiej, do algorytmów dostępnych w aplikacji *Machine Learning Studio* znajdującej się na platformie *Microsoft Azure*.

1.3. Założenie techniczne

Dane prezentowane w **tabeli 1.1** określają podstawowe założenia techniczne przyjęte w trakcie wykonywania analizy porównawczej. Dane te dotyczą między innymi środowiska, w którym wykonane było doświadczenie. Dodatkowo uwzględniono zestaw danych oraz biblioteki użyte w trakcie tworzenia doświadczenia.

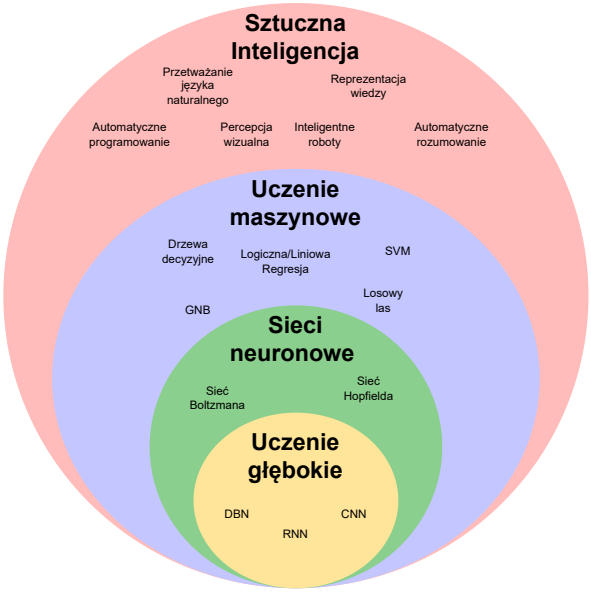
Tabela 1.1. Założenia techniczne pracy dyplomowej
Źródło: Opracowanie własne

Środowisko uruchomieniowe	Machine Learning Studio[1]
Język oporogramowania	Python 3.x
Wykorzystane biblioteki	scikit-learn [**sckit-learn**]
	Numpy [**Harris2019**]
	Pandas [2, 3]
Wykorzystane dane	CICDS2017 [4]

2. Sztuczna inteligencja

Według słownika *Oxford English Dictionary* słowo ”**inteligencja**” oznacza zdolność do rozumienia, a analizy i dostosowania się do zmian[5].

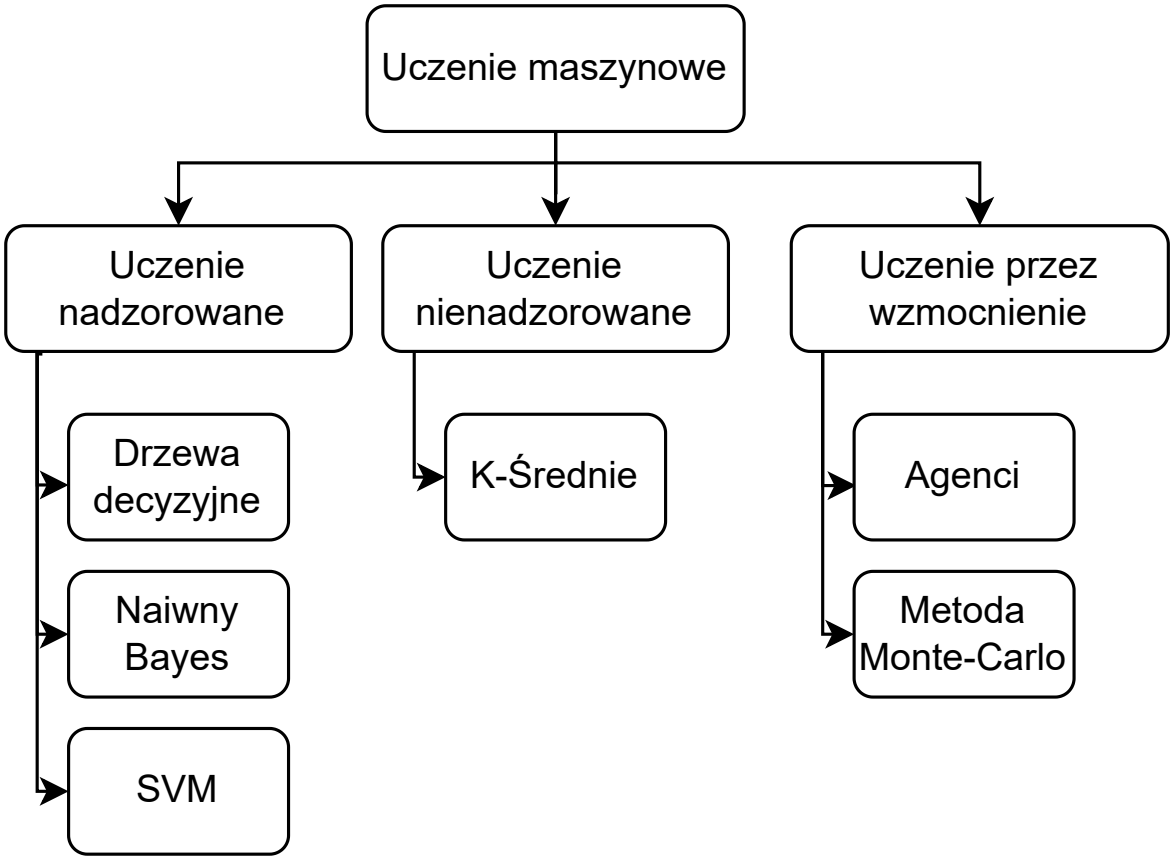
Sztuczna inteligencja(*ang. Artificial Intelligence*) (**AI**) jest wykorzystywana na wiele sposobów podczas prowadzenia badań naukowych: od stawiania hipotez oraz budowania twierdzeń matematycznych, tworzenia i monitorowania badań, zbierania danych i wielu innych czynności towarzyszącymi podczas badań. Najpopularniejszymi zastosowaniami jest między innymi uczenie nienadzorowane oraz wykrywanie anomalii[6, 7]. Schemat podziału sztucznej inteligencji pokazano na **obrazie 2.1**.



Rys. 2.1. Graficzne przedstawienie podziałów sztucznej inteligencji
Źródło: [8]

2.1. Uczenie maszynowe

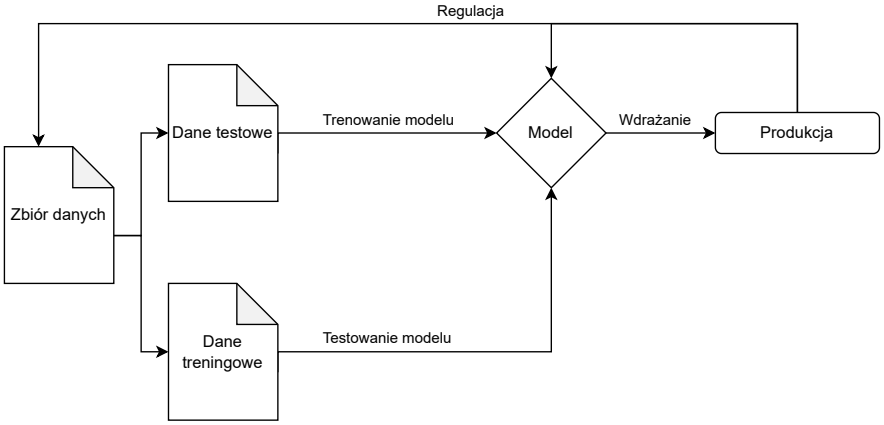
Uczenie maszynowe(*ang. Machine Learning*) (**ML**) jest to dziedzina nauki nad algorytmami oraz modelami statystycznymi, które mogą być wykorzystywane do specyficznych zadań na przykład klasyfikacji, rozpoznawania obrazów bądź mowy, a dodatkowo nie są zaprogramowane specyficznie pod konkretne zadanie, a jedynie pod grupę zadań tak jak pokazano na **obrazie 2.2**. Dlatego też nie ma jednego najlepszego rozwiązania, które można wykorzystać w każdym przypadku. Wykorzystanie konkretnego algorytmu determinuje typ zadania jaki ma być rozwiązany.



Rys. 2.2. Podział uczenia maszynowego
Źródło: [7]

2.1.1. Uczenie nadzorowane

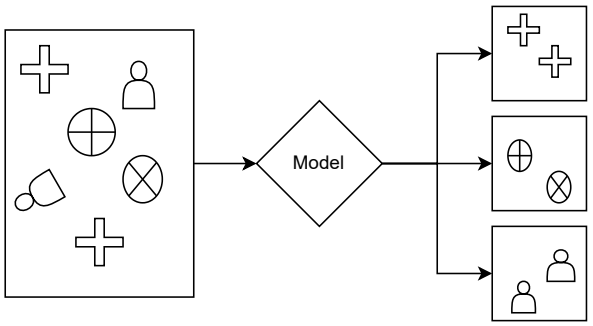
- w trakcie tego uczenia stosuje się zbiór posiadający etykiety. Model uczy się przyporządkowywać określone cechy do konkretnych kategori. Dane wejściowe dzielone są na dane treningowe i dane testowe. Zbiór treningowy jest wykorzystywany do trenowania modelu, a zbió testowy do sprawdzenia rezultatu, na bazie którego może nastąpić korekta uczeniam zilustrowano to **obrazem 2.3**. Algorytmy uczenia nadzorowanego można zastosować między innymi do weryfikacji ruchu sieciowego w celu określenia czy ruch bezpieczny, przez co można to zastosować w systemach wykrywania intruzów(*ang. Intrusion Detection System*) (**IDS**). Algorytmy wchodzące w skład uczenia nadzorowanego to między innymi klasyfikacja naiwna bayesa, drzewa decyzyjne, maszyny wektorów nośnych[6, 7].



Rys. 2.3. Uczenie nadzorowane
Źródło: [7]

2.1.2. Uczenie nienadzorowane

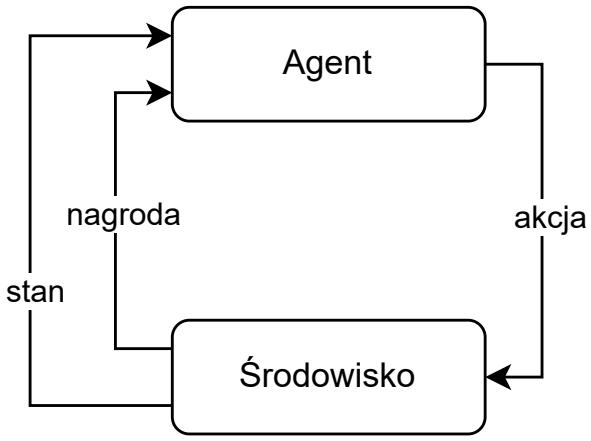
- w tym przypadku nie wykorzystuje się zbioru oznaczonego, algorytm sam próbuje odkryć prawidłową odpowiedź. Dzieję się tak, w danych, których nie da się nazwać albo doprecyzować. Wykorzystuje się to do między innymi detekcji anomalii, co pozwoli do na przykład wykrycia zbyt dużego zużycia prądu w pokoju domu studenckiego dzieki czemu uda się wyłapać nieautoryzowaną koparkę kryptowalut. Dodatkowo można wykorzystać je do szukanai wzorców, albo zarządzania magazynem. W skład takich algorytmów wchodzi: K-średnie, klasteryzacja. Schemat uczenia nienadzorowanego poprzez klasteryzację jest pokazany na **obrazie 2.4**.



Rys. 2.4. Uczenie nienadzorowane
Źródło: Opracowanie własne

2.1.3. Uczenie przez wzmocnienie

- jest to uczenie poprzez nagradzanie dobrych rozwiązań, a karanie złych, potocznie mówiąc jest to metoda ”*kija i marchewki*”. Wykorzystywana w trenowaniu pojazdów autonomicznych pozwala na nagradzanie pojazdów za wybór lepszych tras przykładowo za wybór dróg asfaltowych zamiast polnych. Skupia się w dużym stopniu na agencie i jego decyzjach w danym środowisku co pokazano na **schemacie 2.5**. Należy do jednych z trzech głównych paradygmatów obok uczenia nadzorowanego i nienadzorowanego.



Rys. 2.5. Uczenie przez wzmocnienie
Źródło: [7]

2.1.4. Uczenie częściowo nadzorowane

- do trenowania takich modeli stosuje się niewielkie zbiory oznaczone, oraz większe zbiory nieoznaczone, dzięki którym można próbować rozpoznać rozległe zbiory danych na podstawie pewnych cech wspólnych. Stosuje się to ze względu na mnogość danych na świecie, których opisanie byłoby niemożliwe oraz albo zbyt kosztowne. Przykładowo można znaleźć zastosowanie tych algorytmó w bankowości albo klasyfikowaniu stron internetowych poprzez wyszukiwanie treści na stronie i kategoryzowaniu ich[9]. Jest to połączenie uczenia nadzorowanego i nienadzorowanego[7].

2.2. Sieć neuronowa

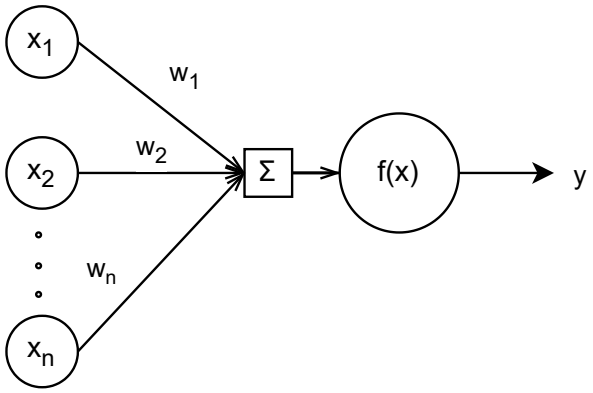
Ludzki mózg jest najbardziej złożonym organem znany ludziom. Badacze zainspirowani jego strukturą składającą się z połączonych ze sobą komórek neuronowych, które przetwarzają równolegle wiele informacji, próbują przenieść pewien poziom inteligencji do komputerów. Przykładem tego jest wiele algorytmów, wchodzących w skład sztucznych sieci neuronowych(*ang. artificial neural network*) (**ANN**), między innymi sieci Kohonena, sieci Hopfielda, sieci konwolucyjne. Sieci te próbują w pewien sposób odwzorować próbę na przykład klasyfikacji danych przez jednostkę wzorowaną na ludzkim mózgu, pomimo tych osiągnięć symulacja ludzkiej świadomości oraz emocji wciąż jest jedynie w sferach fantazji naukowych[**Wang2003**].

Sieć neuronowa jest zbudowana z połaczonych ze sobą warstw neuronów tak jak na **rysunku 2.7**, które w pewien sposób mają wykonać zadania uczenia maszynowego. Najprostszym przykładem sieci neuronowej jest pojedynczy neuron, który może służyć do prostych zadań klasyfikacyjnych: **schemat 2.6**. Sieć ta potrafi się dostosowywać do danych wejściowych tak aby uzyskać odpowiedni wynik, wykonuje w tedy proces uczenia stosując do tego na przykład algorytm wstecznej propagacji wag. W zależności od problemu istnieje wiele różnych sieci, które można zastosować. Jednym z trudniejszych rzeczy w doborze sieci jest dobór warstw ukrytych oraz ilości neuronów, ponieważ w tym celu twórca może opierać się jedynie na własnej wiedzy i doświadczeniu. Podstawy teorii sieci neuronowych zostały stworzone w połowie XX wieku. Złota era uczenia maszynowego rozpoczęła

się dopiero na początku XXI wieku, kiedy to jednocześnie pojawiły się takie trendy jak: Big Data, redukcja kosztów obliczeń równoległych, oraz pierwsze badania nad głębokimi sieciami neuronowymi(*ang. Deep Neural Network*) (**DNN**). Największe zastosowanie DNN miało miejsce dopiero w ostatniej dekadzie kiedy to pojawiły się:

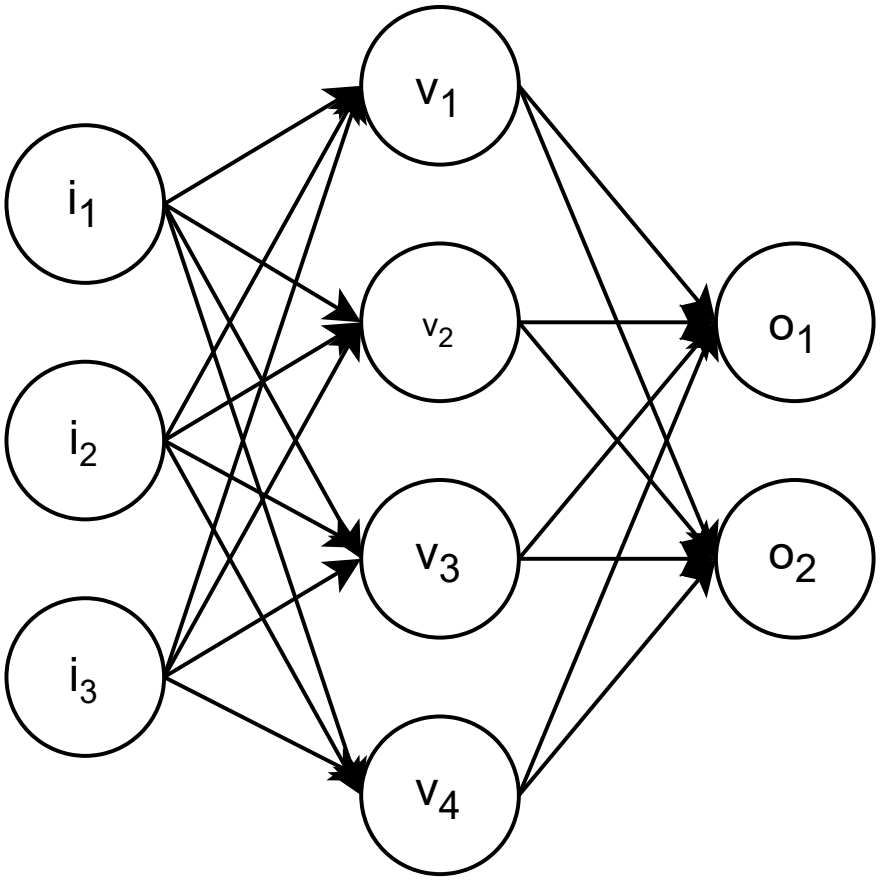
– **Google Braine** - grupa badawcza założona w 2011 roku, zajmująca się badaniami nad sztuczną inteligencja

– **DeepFace** - rozwiązanie stworzone przez firmę Facebook w 2014 roku, służące do rozpoznawania twarzy na zdjęciu [10, 11].



Σ: sumator
$f(x)$: funkcja aktywacyjna
$w_x \; \forall x \; \in [1, 2, ..., n]$: wagi
$x_x \; \forall x \; \in [1, 2, ..., n]$: wejścia

Rys. 2.6. Schemat neuronu
Źródło: Opracowanie własne



$i_x \; \forall x \; \in [1, 2, 3]$: dane wejściowe
$v_x \; \forall x \; \in [1, 2, 3, 4]$: neurony w warstwie ukrytej
$o_x \; \forall x \; \in [1, 2]$: dane wyjściowe

Rys. 2.7. Schemat sieci neuronowej
Źródło: Opracowanie własne

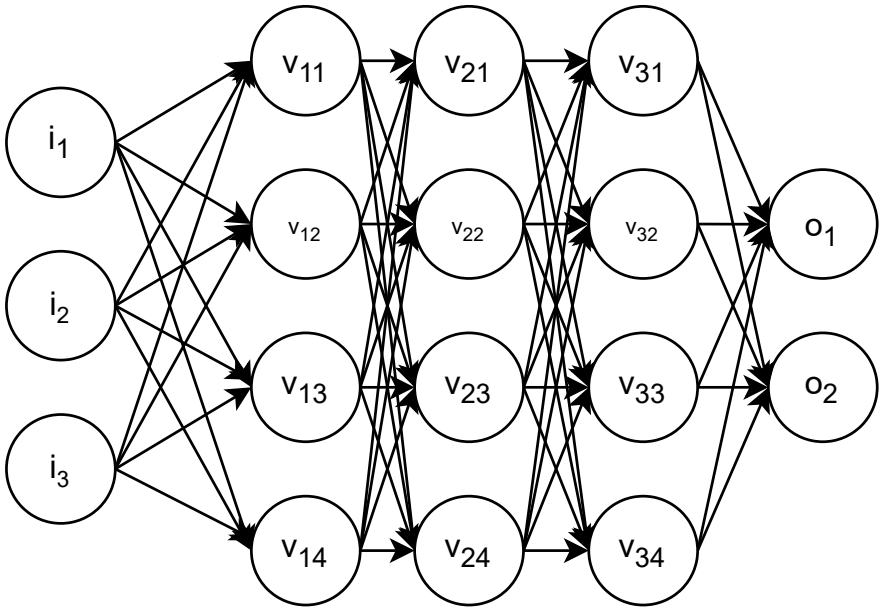
Sieci neuronowe możemy podzielić na wiele rodzajów, do których możemy zaliczyć między innymi:

- **perceptron** - jest to najstarszy przykład sieci neuronowej złożonej z jednego perceptronu (neuronu). Można je zastosować w problemach klasyfikacji;
- **sieci wielowarstwowe perceptronowe** - jest to sieć złożona z wielu warstw połączonych ze sobą neuronów, najprostszy model sieci zaprezentowany na **rysunku 2.7**. Składa się z warstwy wejściowej, warstw (jednej bądź wielu) ukrytych oraz warstwy wyjściowej. W neurony w tej sieci w porównaniu do perceptronów, mają funkcję aktywacyjną sigmoidalną, ze względu na rozwiązywanie problemów nieliniowych (posiadających więcej rozwiązań niż dwa 0/1). Można je zastosować na przykład do klasyfikacji danych;
- **sieci konwolucyjne (CNN)** - są to sieci służące do rozpoznawania obrazów, nazwa wzięła się od wykonywanej na obrazie operacji konwolucji (splotu). Sieci te posiadają dodatkowe warstwy konwolucyjne oraz spłaszczania, które pozwalają zamienić reprezentację obrazu w pojedyncz;

– **sieci rekurencyjne** - charakteryzują się pętlą zwrotną w warstwie ukrytej. Mogą być wykorzystane do generowania tekstu, tłumaczen maszynowych, a także na przykład przewidywania cen rynkowych;
– **sieci samoorganizujące się** - wykorzystuje uczenie nienadzorowane oraz. Składają sie jedynie z warstwy wejściwej i wyjściowej. Zaś cechą charakterystyczną jest to, że neurony określające podobne klasy znajdują się obok siebie. Sieci te mogą być wykrozystywane do podziału klientów na odpowiednie grupy bądź do wskazania jakim klientom zaproponować karty kredytowe [12, 13].

2.2.1. Głębokie uczenie

Jest to podkategoria uczenia maszynowego polegająca na tworzeniu wielowarstwowych sieci neuronowych. W porównaniu do podstawowych sieci neuronowych potrzebuje ogromnych zbiorów danych do utworzenia modelu predykcyjnego. Potrzebuje również dużo więcej mocy obliczeniowej przez wzgląd na ilość warstw ukrytych, których może być dużo więcej, przykładem najprosztszej sieci głębokiej jest **obraz 2.8**.



$i_x \; \forall x \; \in [1, 2, 3]$: dane wejściowe
$v_x \; \forall x \; \in [11, 12, 13, 21, 22, 23, 31, 32, 33]$: neurony w warstwie ukrytej
$o_x \; \forall x \; \in [1, 2]$: dane wyjściowe

Rys. 2.8. Schemat prostej głębokiej sieci neuronowej
Źródło: Opracowanie własne

Warstwa wyjściowa DNN może dostarczać dane róznego formatu, może to być na przykład, tekst, liczba bądź dźwięk. Posiada również bardzo dużo zastosowań w któych skład wchodzi generowanie treści, Deepfake, analiza obrazów, wskazywanie obiektów na obrazach, projektowanie leków, czatboty. Jest to udoskonalenie podstawowych sieci neuronowych. Tak więc częśc typów sieci opisanych w **sekcji 2.2** będzie odnosić się do głębokich sieci neuronowych, należy do nich CNN[**MicrosoftDepp2023**].

3. Klasyfikacja danych

Klasyfikacja jest to próba rozpoznania obiektów na bazie ich cech. Jest to jedna z pierwszych rzeczy jaką uczą się niemowlęta, zaczynając od rozpoznania rodziców, próby rozróżnienia kształtów, kolorów, rzeczy. W otaczającym świecie istnieje wiele mechanizmów mających sklasyfikować rzeczy. Należą do nich katalogi biblioteczne, klasyfikacja trunków, kaw, pojazdów, produktów spożywczych i wielu innych rzeczy. Człowiek od zawsze próbuję skategoryzować i uporządkować posiadaną wiedzę w zbiory ułatwiające obcowanie z tą wiedzą.

W uczeniu maszynowym klasyfikacja jest to metoda uczenia nadzorowanego, podczas której model próbuje przewidzieć etykietę obiektu na podstawie jego cech. Proces uczenia modelu klasyfikacji jest oparty o dwa zbiory, treningowy i testowy, z czego zbiór treningowy powinien być mniejszy od zbioru testowego. Po udanym procesie trenowania modelu, następuje proces testowania na podstawie któego wylicza się odpowiednie metryki pozwalające na ewaluacje modelu. W pracy magisterskiej wykorzystano zbor danych, który posiada dwa typy etykiet [0, 1], dlatego też na tej podstawie zbudowano macierz pomyłek.

3.1. Metryki

W trakcie ewaluacji algorytmu służącego do klasyfikacji wykorzystuje się metryki bazujące na macierzy pomyłek, która została opisana w **tabeli 3.1**

Tabela 3.1. Macierz pomyłek
Źródło: Opracowanie własne

		Prawdziwe wartości	
		1	**0**
Przewidziane wartości	**1**	TP	FN
	0	FP	TN

- **TP** - prawdziwie pozytywny
- **FN** - fałszywie negatywny
- **FP** - fałszywie pozytywny
- **TN** - prawdziwie negatywny

3.1.1. Dokładnośc

$$\frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

Stosunek wszystkich dobrze oznaczonych obiektów do liczby wszystkich prób.

3.1.2. Precyzja

$$\frac{TP}{TP + FP} \tag{3.2}$$

Stosunek poprawnie wybranych obiektów klasy "*I*", do wszystkich wybranych obiektów tej klasy.

3.1.3. Czułość

$$\frac{TP}{TP + FN} \tag{3.3}$$

Stosunek poprawnie sklasyfikowanych obiektów klasy "*I*", do wszystkich obiektów, które powinny być w tej klasie.

3.1.4. F1

$$\frac{2 * x * y}{x + y} \tag{3.4}$$

Jest średnia harmoniczna precyzji (x) i czułości (y);

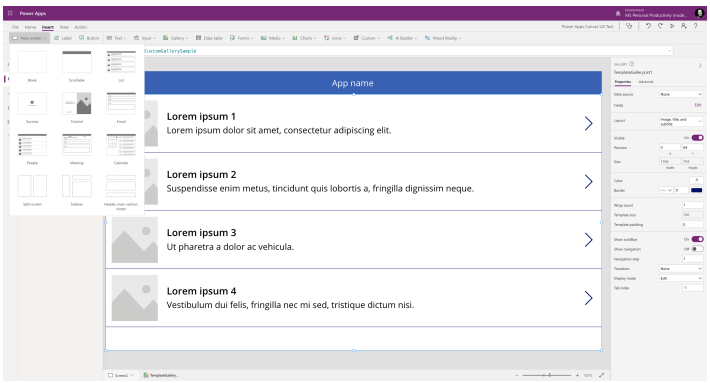
3.1.5. AUC

AUC - (*Area Under Roc Curve*) - Jest to pole pod krzywą ROC pokazuje sprawność klasyfikatora. Im wyższa wartość AUC tym lepiej. Wynik AUC jest z zakresu <0, 1>:

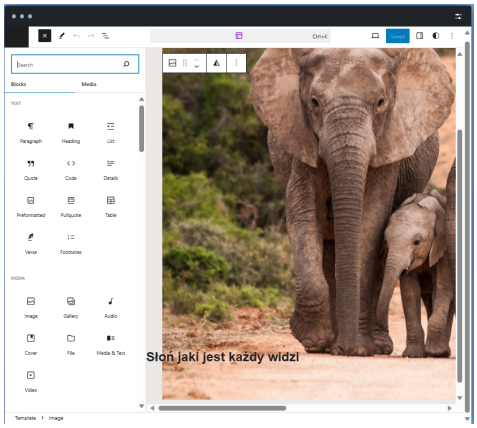
- AUC = 1 - klasyfikator idealny,
- AUC = 0,5 - klasyfikator losowy,
- AUC < 0,5 - klasyfikator gorszy niż losowy[14].

4. Podejście low-code/no-code

Low-Code oraz no-code to nowe podejście skupiające się na umożliwieniu tworzenia programów w sposób nie wymagający znajomości języka oprogramowania. Podejście to ma pozwolić osobom nie będącymi programistami na tworzenie aplikacji biznesowych. Ma to zwiększyć tempo tworzenia rozwiązań biznesowych, na które zapotrzebowanie wciąż rośnie. Jednakże podejście to jest stosounkowo świeżym podejściem, którego początki można było obserwować w codziennym życiu na przykład podczas tworzenia stron internetowych korzystając z narzędzi takich jak *Wordpress*, *Joomla*, *Wix*. Narzędzia te umożliwiają w łatwy sposób tworzyć stronę z tak zwanych kafelków, które umieszczone w odpowiednim miejscu były odpowiedzialne za jedną konkretną rzecz[15, 16, 17]. Przykład na **obrazie 4.1**, **4.2**.



Rys. 4.1. PowerApps od Microsoft
Źródło: [18]



Rys. 4.2. Wordpress.com
Źródło: [19]

Coraz większa popularnością cieszą się platformy low-code(*ang. Low-code Development Platforms*) (**LCDPs**) dostarczane między innymi przez Google, Microsoft, Amazon pozwalają one na tworzenie wysoko skalowalnych rozwiązań przy niewielkim albo i żadnym nakładzie programowania. Ma to umożliwić osobom z niewielkim doświadczeniem w programowaniu, na szybkie wdrożenie oraz tworzenie niezawodnego oprogramowania. Twórcy platform oferują również korzystającym zmniejszenie ilości pracy potrzebnej do wdrożenia albo rozwijania kolejnych funkcjonalności[20, 21].

4.1. Platformy

LCDP udostępniane twórcom aplikacji, umożliwiają skalowalność rozwiązań tworzonych na własne potrzeby. Dodatkowo są popularne przy tworzeniu aplikacji typu "*aplikacja jako usługa*" (*ang. Software-as-a-Service*) (SaaS), które opłacane są tylko za stopień ich użycia, co w niektórych przypadkach może się okazać dużo bardziej opłacalne niż utrzymywanie swoich rozwiązań serwerowych. Dzięki takim rozwiązaniom wiele małych firm będzie mogło pozwolić sobie na tworzenie i utrzymywanie dostosowanych rozwiązań opartych o ekosytem *Microsoft365* / *Google Workspace*.

4.1.1. Microsoft PowerApps

Jest to platforma programistyczna umożliwiająca tworzenie niestandardowych aplikacji dla rozwiązań biznesowych. Umożliwia ona tworzenie aplikacji opartych o różnorakie źródła danych do których należą między innymi: SQL Server, SharePoint, Dynamics 365. Dodatkową zaletą tego rozwiązania jest tworzenie aplikacji responsywnych, działających dobrze na wielu rodzajach urządzeń. Dodatkowow platforma ta pozwala tworzyć trzy typy aplikacji przy braku konieczności kodowania[22]

- **Kanwa** - jest to typ aplikacji oparty o model danych znajdujący się na przykład w Excelu. Aplikację tego typu tworzy się za pomcą przesuwanych kafelek, a proces przypomina po trochu robienie prezentacji przy użyciu aplikacji Powerpoint, co umożliwia pełną dowolność w tworzonym interfejsie graficznym[23].
- **Oparte na modelu** - w ramach korzystania z usługi Microsoft Dataverse można wygenerować aplikacje bazujące na danym modelu danych, przez co użytkownicy otrzymują produkt ułatwiający im analizę danych[24].
- **Karty** - są to uproszczone aplikacje, które można dodać do usługi Micrsoft Teams w określonym biznesowym celu. Dużą zaletą tego rozwiązania jest możliwość korzystania z źródeł danych, dzięki czemu poszczególne karty mogą odpowiadać za jedno zadanie biznesowe[25].

4.1.2. Amazon QuickSight

Jest to rozwiązanie firmy Amazon, które umożliwia firmom dostarczanie rozwiązań z zakresu analityki biznesowe(*ang. business intelligence*) (BI) dzięki interaktywnym pulpitom korzystającym z jednego źródła prawdy. Dodatkowo korzystanie z interaktywnych formularzy, raportów oraz zapytań w języku naturalnym pozwala interesariuszom otrzymać możliwość korzystania z jednolitych rozwiązań opartych o różne modele danych[26].

4.1.3. Google AppSheet

Platforma AppSheet od firmy Google umożliwia tworzenie aplikacji mobilnych oraz desktopowych bez użycia kodu. Firma wskazuje na możliwości integracyjne z różnymi dostawcami danych, do których należą między innymi Microsoft, Dropbox, a także wbudowaną integrację z aplikacjiami Google Workspace do któych należą Gmail, Sheets oraz Spaces. Platforma pozwala również na tworzenie automatycznych botów, które wykonują zadania w oparciu o bodźce zewnętrzne bądź wewnętrzne. Narzędzie pozwala w prosty sposób na tworzenie szybkich rozwiązań biznesowych w oparciu o ekosystem firmy Google[27].

5. Microsoft Azure

6. Opis doświadczenia

7. Analiza porównawcza

8. Perspektywy rozwoju

Wykaz rysunków

Wykaz tabel

Bibliografia

[1] Microsoft. „*Microsoft Machine Learning Studio (classic)*". 2022. URL: *https://studio.azureml.net/* (term. wiz. 2023-09-01).

[2] The Pandas development team. „*pandas-dev/pandas: Pandas*". Lut. 2019. DOI: *9.5281/zenodo.3509134*. URL: *https://doi.org/9.5281/zenodo.3509134*.

[3] Wes McKinney. „*Data Structures for Statistical Computing in Python*". W: *Proceedings of the 9th Python in Science Conference*. 2010, s. 56–61. DOI: *10.25080/majora-92bf1922-00a*.

[4] UNB. „*CICIDS2017 | Kaggle*". URL: *https://www.kaggle.com/datasets/cicdataset/cicids2017* (term. wiz. 2023-09-01).

[5] Oxford University Press. „*intelligence, n., sense 1*". W: *Oxford English Dictionary*. Lip. 2023. DOI: *10.1093/OED/3757635879*.

[6] „*Artificial Intelligence in Science*". OECD, czer. 2023. DOI: *10.1787/a8d820bd-en*.

[7] Batta Mahesh. „*Machine Learning Algorithms-A Review*". W: *International Journal of Science and Research* (2018). ISSN: 2319-7064. DOI: *10.21275/ART20203995*.

[8] Satavisa Pati. „*The Difference Between Artificial Intelligence and Machine Learning*". W: *Emerj* Ml (2018), s. 3–8.

[9] Chun Zhang i in. „*Semi-supervised behavioral learning and its application*". W: *Optik* 127.1 (2016), s. 376–382. ISSN: 00304026. DOI: *10.1016/j.ijleo.2015.10.089*.

[10] Robert Koch. „*History of Machine Learning – A Journey through the Timeline*". 2022. URL: *https://www.clickworker.com/customer-blog/history-of-machine-learning/* (term. wiz. 2023-09-02).

[11] Alexander L. Fradkov. „*Early history of machine learning*". W: *IFAC-PapersOnLine*. T. 53. 2. Elsevier, sty. 2020, s. 1385–1390. DOI: *10.1016/j.ifacol.2020.12.1888*.

[12] Austin Pollard. „*What are neural networks?*" 1990. DOI: *10.1108/eb007822*. URL: *https://www.ibm.com/topics/neural-networks*.

[13] Karolina Bartos. „*SIEĆ SOM JAKO PRZYKŁAD SIECI SAMOORGANIZUJACEJ SI*□". W: (2012). ISSN: 1507-3866.

[14] Algolytics. „*Jak ocenić jakość i poprawność modeli klasyfikacyjnych ? Cz*□*ść 4- Krzywa ROC*". URL: *https://algolytics.pl/tutorial-jak-ocenic-jakosc-i-poprawnosc-modeli-klasyfikacyjnych-czesc-4-krzywa-roc/* (term. wiz. 2023-09-04).

[15] Wordpress. „*WordPress.com: Build a Site, Sell Your Stuff*". 2023. URL: *https://wordpress.com/* (term. wiz. 2023-09-04).

[16] JoomlaORG. „*Joomla! - Content Management System to build websites*". 2021. URL: *https://www.joomla.org/%20https://www.joomla.org/about-joomla.html* (term. wiz. 2023-09-04).

[17] Wix. „*Free website builder | Create a free website*". 2016. URL: *https://www.wix.com/* (term. wiz. 2023-09-04).

[18] Microsoft. „*PowerApps*". 2023. URL: *https://guidedtour.microsoft.com/guidedtour/scenarios/power-apps/2.2.png* (term. wiz. 2023-09-04).

[19] Wordpress. „*Playground Demo*". 2023. URL: *https://developer.wordpress.org/playground/demo/* (term. wiz. 2023-09-04).

[20] Alexander C. Bock i Ulrich Frank. „*Low-Code Platform*". W: *Business and Information Systems Engineering* 63.6 (grud. 2021), s. 733–740. ISSN: 18670202. DOI: *10.1007/S12599-021-00726-8/FIGURES/1*.

[21] Martin Hirzel. „*Low-Code Programming Models*". W: (maj 2022). arXiv: *2205.02282*.

[22] Microsoft. „*Co to jest usługa Power Apps? - Power Apps | Microsoft Learn*". URL: *https://learn.microsoft.com/pl-pl/power-apps/powerapps-overview* (term. wiz. 2023-09-05).

[23] Microsoft. „*Omówienie tworzenia aplikacji kanw - Power Apps | Microsoft Learn*". URL: *https://learn.microsoft.com/pl-pl/power-apps/maker/canvas-apps/getting-started* (term. wiz. 2023-09-05).

[24] Microsoft. „*Omówienie tworzenia aplikacji opartej na modelu z Power Apps - Power Apps | Microsoft Learn*". URL: *https://learn.microsoft.com/pl-pl/power-apps/maker/model-driven-apps/model-driven-app-overview* (term. wiz. 2023-09-05).

[25] Microsoft. „*Omówienie kart dla usługi Power Apps - Power Apps | Microsoft Learn*". URL: *https://learn.microsoft.com/pl-pl/power-apps/cards/overview* (term. wiz. 2023-09-05).

[26] AmazonQuickSight. „*Business Intelligence Service – Amazon QuickSight – AWS*". URL: *https://aws.amazon.com/quicksight/* (term. wiz. 2023-09-05).

[27] GoogleAppSheet. „*Google AppSheet | Build apps with no code*". URL: *https://about.appsheet.com/home/* (term. wiz. 2023-09-05).